Collecteur & fichier de configuration *netmet.conf*

1 Avant de commencer…

Avant de commencer la description détaillée de la syntaxe du fichier de configuration *netmet.conf*, il est sans doute intéressant de comprendre l'intérêt de ce fichier et de voir dans quel contexte il s'utilise.

Le collecteur netMET est constitué de 2 processus : netMETcll et netMETacc.

1.1 Les processus du collecteur

1. netMETcll : un processus de réception des paquets UDP NetFlow en provenance directe d'un routeur ou du duplicateur de flows de netMET (netMETdup).

netMETcll est capable :

- de décoder les paquets NetFlow selon les formats 1, 5 et 7,
- de recevoir de paquets NetFlow en provenance de plusieurs routeurs ou duplicateurs différents,
- d'exclure des flows reçus qui correspondent à une détection de flux "non souhaitée" pour la métrologie à appliquer (ce qui revient à pourvoir garder uniquement les flows qui nous intéressent),
- d'agréger "à la volée" tous les flows en provenance/destination d'une interface particulière du routeur avec une adresse IP symbolique ("virtuelle") : réalisation d'un "trou noir".

Une fois toutes ces opérations appliquées sur les flows NetFlow originels, netMETcll construit des MetFlow (qui sont en fait des NetFlow épurés avec application de règles précédentes) puis les passe au processus netMETacc.

2. netMETacc : un processus d'accounting de MetFlow. netMETacc réalise l'accounting -c'est à dire la gestion de compteurs de métrologie dédiés- en temps réel en gérant en mémoire des structures de données plus ou moins complexes.

Le fonctionnement entre le processus netMETcll et netMETacc ainsi que la circulation des différentes informations mise en oeuvre sont illustrés par le schéma Figure 1.

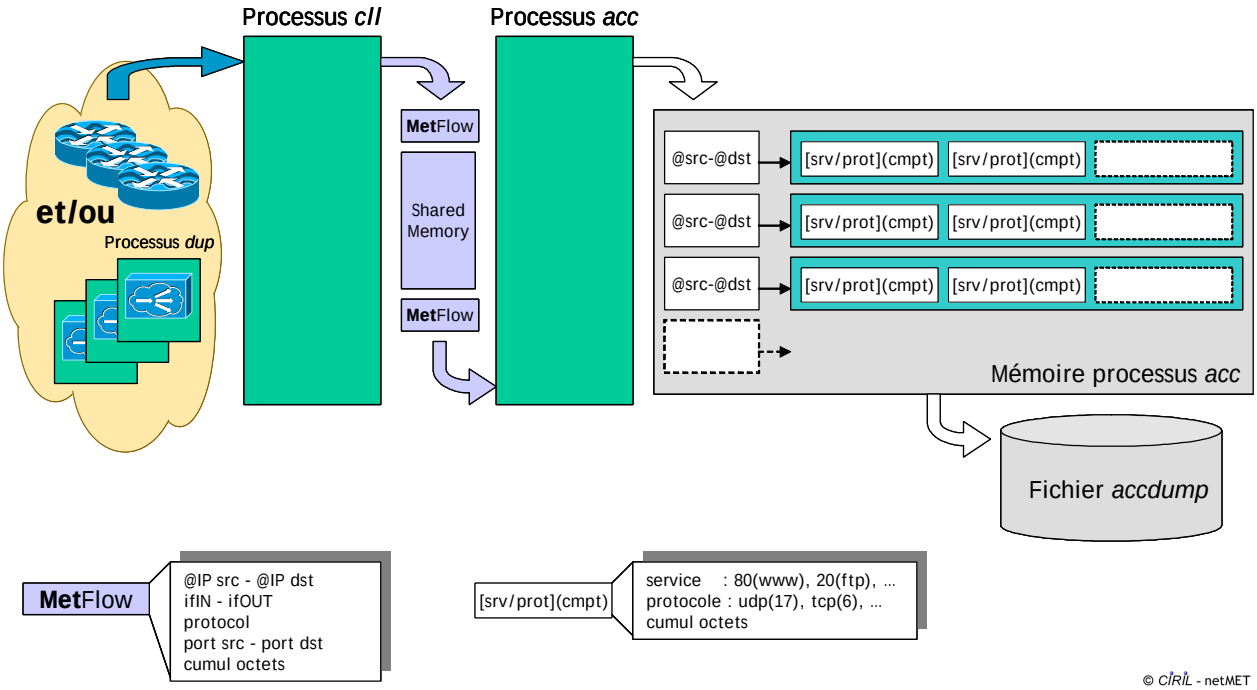

Figure 1 - Fonctionnement de netMETcll et netMETacc

Précisons que le fichier de configuration n'est pris en compte que par le processus netMETcll, mais étant donné qu'il va modifier le comportement de netMETcll, on peut facilement imaginer qu'il influencera également les données (et donc la métrologie) passées à netMETacc.

Faisons maintenant quelques remarques sur l'activation du NetFlow sur un routeur et sur les informations de flows qu'envoie ce routeur vers le collecteur netMET.

1.2 NetFlow ®

Précisions (ou reprécisions) que la technologie NetFlow ® est une technologie Cisco qui est embarquée dans les routeurs et qui, de ce fait, ne peut être exploitée qu'à partir de routeurs de cette marque. Toute la documentation NetFlow Cisco ® est disponible sur le Web http://www.cisco.com, mais les liens suivants sont quelques références plus directes :

- http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm
- http://www.cisco.com/warp/public/732/Tech/netflow
- http://www.cisco.com/univercd/cc/td/doc/pcat/ntfl.htm
- http://www.cisco.com/go/netflow

La Figure 2 représente un routeur à 2 interfaces sur lequel on a activité le *NetFlow* uniquement sur l'interface IF-0 (point rouge). La détection de nouveaux flux ne peut se faire que lorsque que les paquets entrent dans le routeur (quand les paquets sortent du routeur c'est un peu tard :-) !) après quoi les mécanismes d'accélération rentrent en jeu. Le *NetFlow* gère, en interne, un cache des flux détectés qui peuvent être envoyés vers une machine (*netMET* par ex.) dès qu'ils sont invalidés (invalidation = flux terminé). On comprend bien maintenant pourquoi le routeur n'enverra des informations de flux ne concernant que le sens IF-0 -> IF-1 (flèche rouge et sens de la flèche).

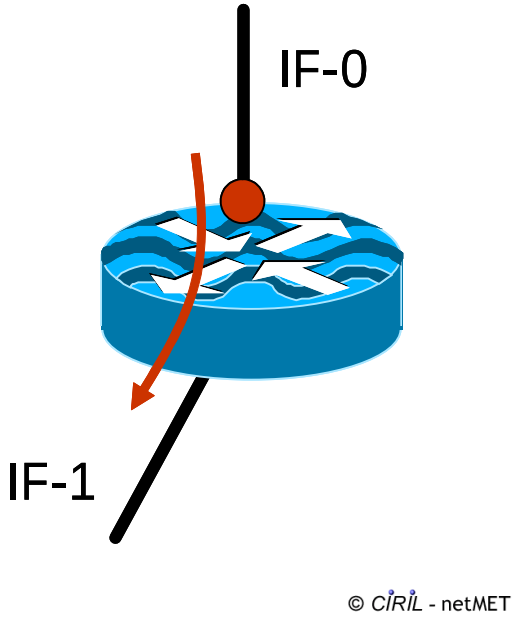

Figure 2 - Routeur 2if, 1 activation

Si l'on souhaite avoir des informations sur les paquets en retour de ceux du sens IF-0 -> IF-1, il faut activer le NetFlow sur l'interface IF-1. La Figure 3 illustre l'activation sur l'interface IF-1. Le fait d'avoir activer le NetFlow sur les interfaces IF-0 et IF-1 permet donc d'avoir les informations de flow dans les sens aller et retour.

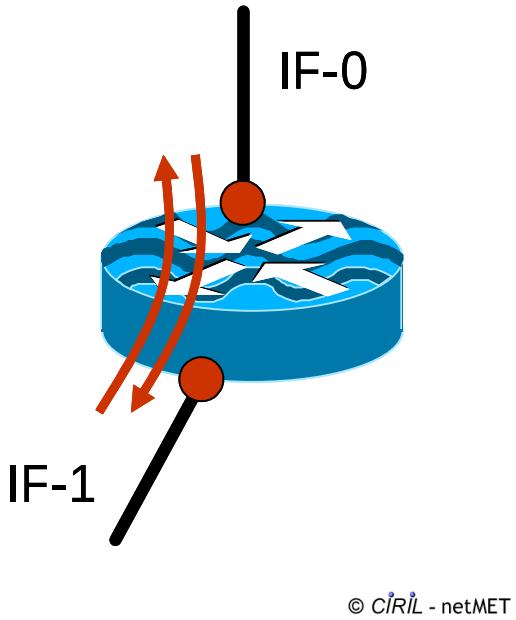

Figure 3 - Routeur 2if, 2 activations NetFlow

On peut bien sûr étendre cet exemple à une routeur à 3 interfaces, comme l'illustre la Figure 4. Seulement le fait d'avoir activer le NetFlow sur l'interface IF-2 donne également des informations de métrologie pour les paquets IF-1 -> IF-2 et IF-2 -> IF-1.

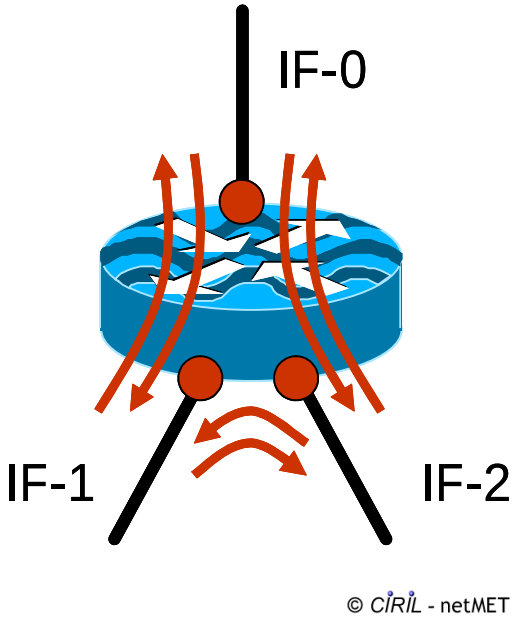

Figure 4 - Routeur 3if, 3 activations NetFlow

1.3 Quelques questions sur le collecteur

Après ces brefs rappels d'architecture netMET et d'activation NetFlow sur un routeur, nous pouvons nous poser quelques questions sur le fonctionnement du collecteur...

- Comment le collecteur sait-il sur quel @IP/port il doit écouter pour collecter les paquets UDP NetFlow ?
- Comment exclure/conserver les flows qui correspondent à des "sens" bien particulier ? Exemple : Ne conserver que IF-0 <-> IF-1 et IF-0 <-> IF-2 (et donc exclure IF-1<->IF-2).
- Comment considérer une interface comme un "trou noir" ? Exemple : dire que tout ce qui entre/sort de l'interface IF-0 correspond à vient/va vers l'Internet.
- Le collecteur n'étant pas dédié à un seul routeur/duplicateur, comment faire pour exprimer les règles précédentes de façons différentes sur les différents routeurs sources ?

Les sections qui font suivre vont répondre une à une à ces questions tout en expliquant la syntaxe du fichier de configuration qui permettra d'exprimer ces règles. De plus, la description de ces règles ne peut pas se faire sans explication sur le collecteur lui-même, c'est donc l'occasion de bien comprendre comment il fonctionne...

2 Emplacement du fichier de configuration netmet.conf

Le fichier de configuration netmet.conf est utilisé, comme nous l'avons déjà dit, par le processus netMETcll du collecteur netMET (collecteur toujours constitué des processus netMETcll et netMETacc). Or sur une machine peuvent cohabiter plusieurs collecteurs et donc plusieurs processus netMETcll, ce qui n'autorise pas l'emploi d'un fichier de configuration unique sous `/etc/netmet.conf`, comme cela est réalisé pour la plupart des services d'une machine UNIX.

La technique employée dans notre cas sera de mettre le fichier *netmet.conf* à "proximité" de l'exécutable netMETcll, de manière à ce que celui soit renseigné sur la configuration à utiliser. Il doit se trouver en fait dans un répertoire etc/ à partir de l'exécutable netMETcll (et donc de netMETacc). La Figure 5 illustre les fichiers et répertoires à mettre en place pour utiliser un nouveau collecteur.

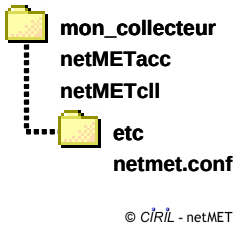

Figure 5 - Collecteur netMet et emplacement du fichier netmet.conf

Remarquons que le répertoire `etc/` peut également se trouver à "proximité" d'un lien symbolique vers les exécutables netMETcll et netMETacc, ce qui permet d'avoir une et une seule instance des exécutables binaires, utilisables pour "activer" plusieurs collecteurs. Cette technique a bien entendu été utilisée dans la distribution de netMET avec ses 4 services : metro, stats, secure24h et secure10m qui correspondent en réalité à 4 collecteurs différents, utilisant pourtant les mêmes binaires. Ce fonctionnement est illustré par la Figure 6.

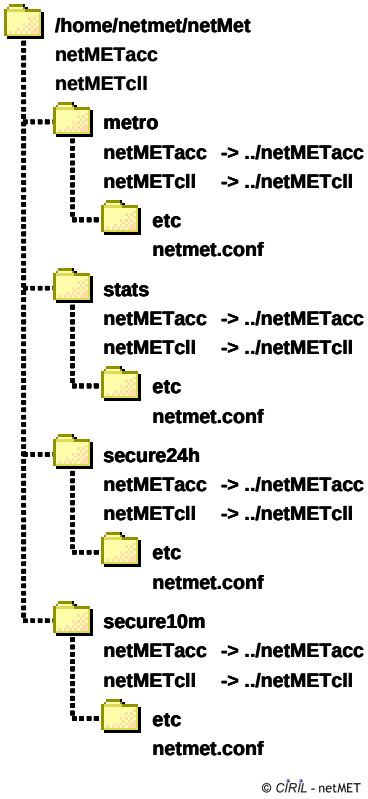

Figure 6 - Les services netMET et emplacements des fichiers *netmet.conf*

3 Syntaxe générale du fichier netmet.conf

Cette partie explique rapidement la syntaxe du fichier de configuration au niveau de l'utilisation des commentaires et de la "grammaire" employée.

- Commentaires :
  - tout ce qui se trouve derrière un # est considéré comme commentaire jusqu'à la fin de ligne (commentaire "à la perl"),
  - tout ce qui se trouve entre deux /* ... */ est également considérer comme commentaire (commentaire "à la C").

- Grammaire :

```
NETFLOW_LISTEN_ADDR_PORT { hhh.hhh.hhh.hhh/pppp }

ggg.ggg.ggg.ggg1 /* router_1 section */
{
  SNMP_READ_COMMUNITY { "community" }

  IF_PROCESSED
   {
     ALL |
     IF_1 <-> IF_2 [, IF_n <-> IF_m ... ]
     /* Where IF_i = "SNMP ifDescr"
           or IF_i =  OTHER */
   }

  [ IF_AGGREGATION
   {
     IF_1 (aaa.aaa.aaa.aaa) [ , IF_n (aaa.aaa.aaa.aaa) ... ]
     /* Where IF_i = "SNMP ifDescr" */
   } ]
}

[ ggg.ggg.ggg.gggi /* router_i section */
{
  ...
} ]
```

Figure 7 - Grammaire employée pour configuration *netmet.conf*

- Remarques :
  - [section] signifie que section est optionnelle,
  - ... signifie que le motif (pattern) précédent peut être répété,
  - " est significatif : le nom de la communauté READ SNMP ainsi que les noms d'interfaces doivent être entourés de " ",
  - cette syntaxe est de type bloc : un bloc commence par un mot clé (ex: NETFLOW_LISTEN_ADDR_PORT, IF_PROCESSED, ...) puis est délimité par des { },
  - les majuscules/minuscules sont significatifs,
  - les sauts de lignes ne sont pas significatifs,
  - les séparateurs blancs espaces et tabulations ne sont pas significatifs,
  - le séparateur de listes est la virgule : ",",
  - voir les exemples dans la suite...

3.1 La clause *NETFLOW_LISTEN_ADDR_PORT*

- Obligatoire
- Niveau : général.
- Nombre : 1.
- Syntaxe :
- NETFLOW_LISTEN_ADDR_PORT { hhh.hhh.hhh.hhh/pppp }

- Paramètres :
    - hhh.hhh.hhh.hhh : adresse IP de l'interface locale de la machine de métrologie vers laquelle sont envoyés les datagrammes UDP NetFlow depuis un routeur directement ou depuis le collecteur netMET. Ex : hhh.hhh.hhh.hhh = 192.168.200.1
    - pppp : port UDP d'écoute des datagrammes UDP NetFlow. Ex : pppp = 8081.

La clause NETFLOW_LISTEN_ADDR_PORT permet d'indiquer au collecteur sur quelle adresse IP locale et quel port il doit écouter les paquets UDP NetFlow. L'intérêt de l'utilisation de ports configurables est de pouvoir démarrer plusieurs collecteurs qui écoutent chacun sur un port spécifique. De même la désignation d'une adresse IP locale permet de spécifier une interface particulière dans le cas d'une machine avec plusieurs cartes réseau.

3.2 *Section routeur : règles appliquées aux flows d'un routeur source*

- Obligatoire.
- Niveau : général.
- Nombre : 1 à n.
- Syntaxe : ggg.ggg.ggg.ggg { router_rules }
- Paramètres :
    - ggg.ggg.ggg.ggg : section pour un routeur susceptible d'envoyer des datagrammes UDP NetFlow depuis son interface ayant comme adresse IP : ggg.ggg.ggg.ggg. Ex : ggg.ggg.ggg.ggg = 192.168.200.254.
    - router_rules : apparaissent ici les caractéristiques et règles à utiliser en cas de réception de paquets NetFlow depuis le routeur ggg.ggg.ggg.ggg. Ces caractéristiques et règles sont les clauses SNMP_READ_COMMUNITY, IF_PROCESSED, IF_AGGREGATION détaillées dans les sections suivantes.

Nous avons déjà vu qu'un collecteur est capable de recevoir des paquets UDP NetFlow depuis des routeurs sources différents soit directement, soit au travers du duplicateur netMET. Nous savons également que des opérations spécifiques sur les flows de routeurs particuliers pourront être appliquées. En fait c'est cette règle qui permet "d'aiguiller" les paquets NetFlow vers la "bonne" section du routeur qui les concerne. Ainsi l'administrateur est sûr que les règles qu'il "programme" dans une section routeur ne seront appliquées que sur les flows provenant de ce routeur particulier.

Pour connaître l'adresse IP source des paquets NetFlow il existe plusieurs solutions :

- demander à son administrateur réseau si il a utilisé la commande 'ip flow-export source ...' sur le routeur, auquel cas il pourra sans doute spécifier l'adresse IP source des paquets UDP,
- ou utiliser la commande 'tcpdump' du système UNIX en spécifiant le port sur lequel sont sensés être exportés les paquets UDP. Ex : "tcpdump -n 'dst port 8081'". ATTENTION, cette technique ne peut s'employer qu'au plus près du routeur (la machine vers laquelle sont exportés directement les NetFlow depuis le routeur) sinon, si l'on se place sur une machine qui se trouve "alimentée" par un duplicateur netMET, l'adresse source obtenue sera celle du duplicateur... or c'est bien l'adresse du routeur qui nous intéresse.

Cette dernière remarque donne l'occasion de préciser que le duplicateur netMET pour propager l'adresse IP source du/des routeur(s) qui a envoyé les NetFlow originels opère une petite modification sur les paquets UDP (un peu à la Garcimore :-) !).

3.3 La clause SNMP_READ_COMMUNITY

- Obligatoire.
- Niveau : section routeur.
- Nombre : 1.
- Syntaxe : SNMP_READ_COMMUNITY { "community" }
- Paramètre :
    - community : mot de passe pour la communauté READ SNMP du routeur ggg.ggg.ggg.ggg. Ex : community = public.

Le collecteur netMET a besoin d'avoir un accès SNMP en lecture sur le(s) routeur(s) source(s) des paquets NetFlow. Cela peut être une contrainte plus ou moins forte pour la mise en place d'un collecteur, néanmoins cette technique permet d'écrire un fichier de configuration très lisible et qui restera cohérente dans le temps.

En effet comme nous l'avons exprimé précédemment, le collecteur doit être capable d'effectuer des opérations sur certains flows (conservation, suppression, agrégation) en fonction de leurs interfaces d'entrée/sortie. L'information d'interface d'entrée/sortie est bien entendu contenue dans les paquets NetFlow mais sous forme d'un numéro d'interface (ex: 22). Or pour une interface physique ou virtuelle ce numéro n'est pas attribué définitivement et peut changer lors d'une réorganisation du châssis (hardware) ou lors d'ajout/suppression d'interfaces virtuelles (software). Dans ce dernier cas notons que le changement de numéro d'interface ne sera effectif qu'au reboot du routeur : le routeur réorganise en effet ses numéros d'interfaces (anciennes et nouvelles crées) au reboot.

Les numéros d'interfaces utilisés dans les paquets NetFlow correspondent aux numéros d'interfaces des OID (Object IDentifier) de la MIB-II SNMP. Ils apparaissent, entre autre, dans l'OID : .iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifDescr.ifIndex qui donne pour l'interface de numéro ifIndex sa description (Ex: "ATM2/0.999-aal5 layer").

Il est clair qu'il est plus facile à l'administrateur netMET de gérer des noms d'interfaces (Ex: "ATM2/0.999-aal5 layer") que des numéros (Ex: 22), d'autant plus que le numéro sont susceptibles de changer alors qu'une interface particulière ne changera jamais de nom. Apparaît donc clairement l'intérêt d'une requête SNMP permettant de construire une correspondance ifDescr <-> ifIndex, et autorisant ainsi l'administrateur à désigner les interfaces du(es) routeur(s) par leur nom et non par des numéros. Les clauses IF_PROCESSED et IF_AGGREGATION utilisent cette notion de nom d'interface.

3.4 La clause IF_PROCESSED

- Obligatoire.
- Niveau : section routeur.
- Nombre : 1.
- Syntaxe : IF_PROCESSED { ALL | IF_1 <-> IF_2 [, IF_n <-> IF_m ...] }
  où : IF_i = "IfDesc SNMP" ou IF_i = OTHER
- Paramètres :
    - ALL : mot clé réservé, doit être utilisé seul : IF_PROCESSED { ALL }
      Signifie que le collecteur netMET va prendre en compte tous les flows provenant du routeur source sans faire de suppressions basées sur les informations d'interfaces d'entrée/sortie de ces flows.
    - IF_i étant soit :
        - "ifDescr SNMP" qui correspond à la description d'une interface physique ou virtuelle du routeur,
        - soit le mot clé réservé OTHER qui correspond à toutes les interfaces sauf celle déclarée dans la même règle. OTHER ne peut s'employer que d'un coté des <->. Ex : OTHER <-> "ifDescr SNMP" ou "ifDescr SNMP" <-> OTHER mais jamais OTHER <-> OTHER.

      L'utilisation de règles avec IF_i indique au collecteur netMET que seuls certains flows doivent être traités : ceux détectés entre IF_o et IF_p sans notion de sens, c'est à dire ceux détectés dans le sens IF_o -> IF_p mais également dans le sens IF_p -> IF_o.

Quelques explications supplémentaires appuyées d'exemples concrets s'imposent pour bien comprendre toutes les subtilités de la syntaxe de la règle IF_PROCESSED. L'exemple illustré par la Figure 8 est le même que celui de la Figure 4 (Avant de commencer...) : nous utilisons un routeur à 3 interfaces sur lesquelles le NetFlow a été activé. Le choix de cet exemple est simple : c'est à partir de n (n>2) interfaces que des problèmes de réceptions de flows non désirés (informations de métrologie entre interface IF_o et IF_p non désirées) se pose et l'étude pour n=3 illustre bien tous les cas possibles.

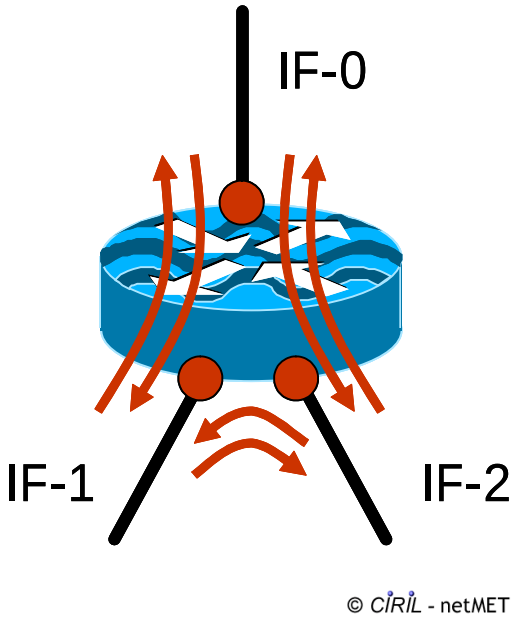

Figure 8 - Routeur 3if, 3 activations *NetFlow*

Nous savons que le fait d'avoir activé le NetFlow sur les 3 interfaces générera des informations de métrologie pour chaque couple d'interfaces et ceci dans les 2 sens. Le collecteur est donc susceptible de recevoir des paquets NetFlow ayant été détectés pour les communications :

- IF-0 -> IF-1
- IF-1 -> IF-0
- IF-0 -> IF-2
- IF-2 -> IF-0
- IF-1 -> IF-2
- IF-2 -> IF-1

REMARQUES :

- dans les schémas qui vont suivre les flèches vertes correspondent aux flows que l'on souhaite conserver et non aux flows que le routeur est susceptible d'envoyer (se reporter à Avant de commencer...),
- comme nous l'avons expliqué dans La clause SNMP_READ_COMMUNITY, le fichier de configuration netmet.conf désigne les interfaces des routeurs par leur description SNMP de la MIB-II standard. Pour connaître ces descriptions, l'administrateur peut utiliser le script perl livré dans distribution netMET sous : ~netmet/netMet/scripts/getIF.sh (se reporter à Tips and Tricks : Choix de la "bonne" interface pour plus de détails). Ce script réalise en fait un snmpwalk sur toutes les interfaces du routeur et affiche le numéro et la description de chaque interface. Ne pas oublier les " " autour des descriptions d'interfaces.

Imaginons que l'application de métrologie a effectivement besoin de tous ces sens : IF-0 <-> IF-1, IF-0 <-> IF-2, IF-1 <-> IF-2 comme dans la Figure 9. Dans ce cas la configuration est très simple car il suffit d'utiliser dans la clause IF_PROCESSED le mot clé ALL qui autorise le traitement de tous les flows quelle que soit l'interface d'entrée/sortie.

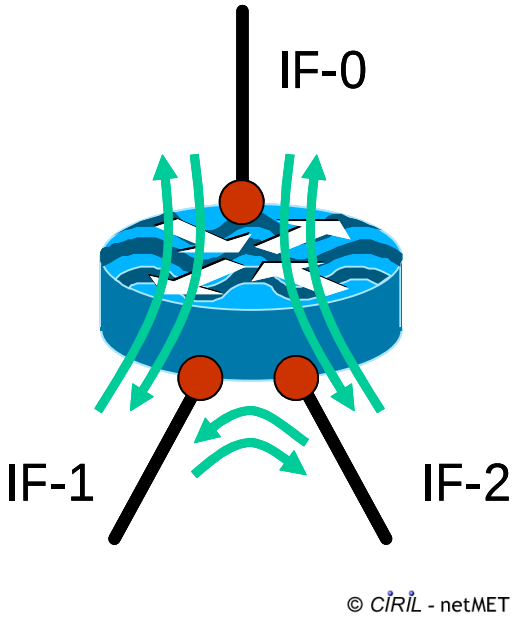

Figure 9 - Utilisation du mot clé ALL

- La configuration à utiliser sera :
  IF_PROCESSED { ALL }
- une configuration équivalente, mais plus longue, pourrait être :

```
IF_PROCESSED
{
   "IF-0" <-> "IF-1" ,
   "IF-0" <-> "IF-2" ,
   "IF-1" <-> "IF-2"
}
```

Maintenant imaginons que l'application de métrologie n'a besoin que des flows entre IF-1 et IF-2 comme l'illustre la Figure 10. La configuration est là encore très simple car elle se résume à exprimer IF-1 <-> IF-2. Remarquons que dans cet exemple l'activation du NetFlow sur l'interface IF-0 n'est pas obligatoire pour obtenir les flows entre IF-1 et IF-2, néanmoins il se peut que le routeur exporte ses flows vers une autre machine de métrologie (soit directement, soit au travers du duplicateur netMET) qui, elle, en a besoin.

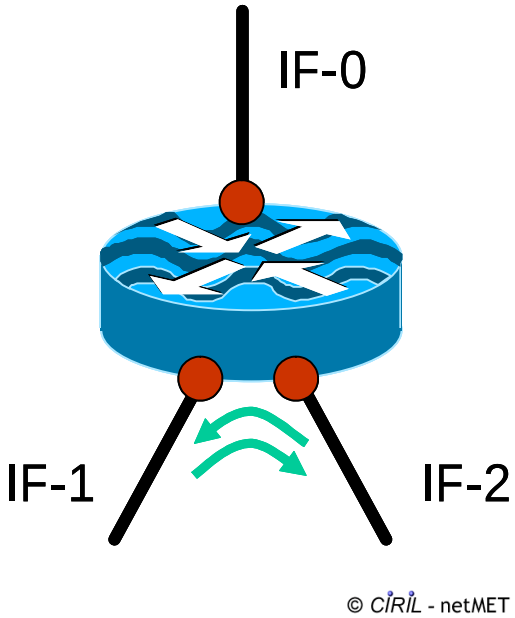

Figure 10 - Utilisation des descriptions d'interfaces : "ifDescr SNMP"

- La configuration à utiliser sera :
```
IF_PROCESSED
{ "IF-1" <-> "IF-2" }
```
- mais comme le sens n'a pas de signification au niveau de la syntaxe, on peut également écrire :
```
IF_PROCESSED
{ "IF-2" <-> "IF-1" }
```

La suite logique est, par exemple, de vouloir conserver les flows entre IF-0 <-> IF-1 et IF-0 <-> IF-2 comme sur la Figure 11. Il est très facile d'exprimer cela à l'aide de l'exemple précédent avec l'ajout d'une règle supplémentaire, mais le mot clé OTHER permet quant à lui de simplifier la configuration obtenue.

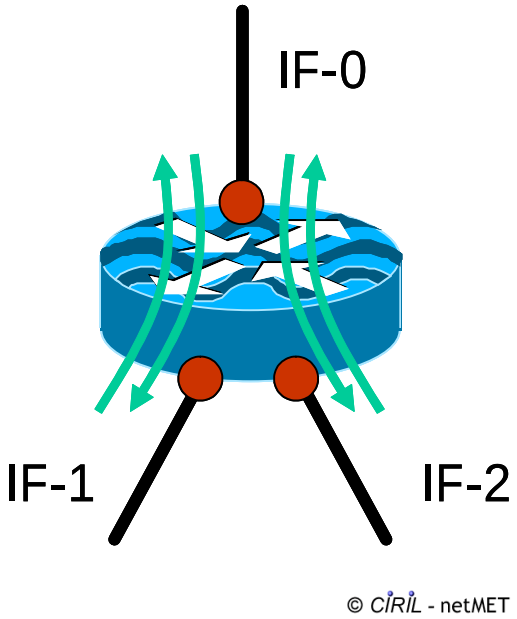

Figure 11 - Utilisation du mot clé OTHER

- La configuration à utiliser sera :
```
IF_PROCESSED
{ "IF-0" <-> OTHER }
```
- sans l'utilisation du mot clé OTHER, la configuration pourrait être :
```
IF_PROCESSED
{
  "IF-0" <-> "IF-1",
  "IF-0" <-> "IF-2"
}
```

Pour finir cette série d'exemples, prenons un routeur à 4 interfaces sur lesquelles le NetFlow a été activé : Figure 12. Ici on souhaite ne conserver que les flows entre IF-0 vers toutes les interfaces et IF-2 <-> IF-3. Pour réaliser cette configuration (le plus simplement possible !) on peut faire un mixe entre le mot clé OTHER et "ifDescr SNMP".

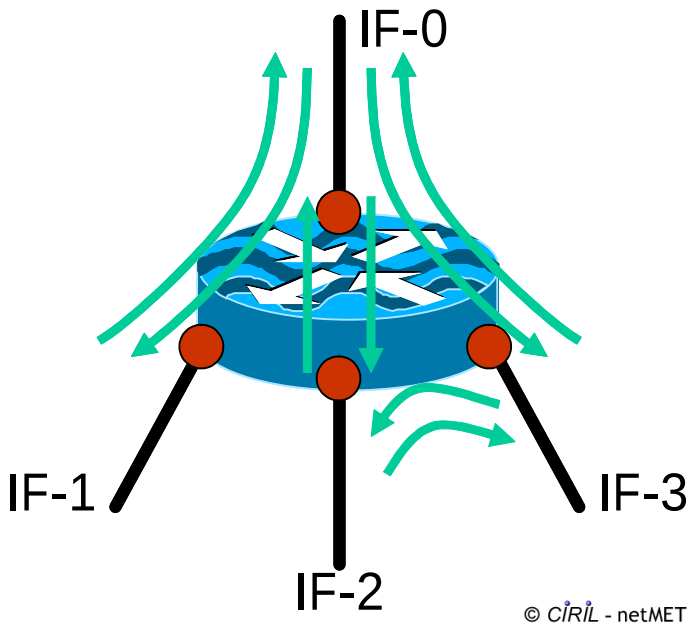

Figure 12 - Mix entre le mot clé OTHER et "ifDescr SNMP"

- La configuration à utiliser sera :

```
IF_PROCESSED
{
    "IF-0" <-> OTHER ,
    "IF-2" <-> "IF-3"
}
```

- sans l'utilisation du mot clé OTHER, la configuration pourrait être :

```
IF_PROCESSED
{
    "IF-0" <-> "IF-1",
    "IF-0" <-> "IF-2",
    "IF-0" <-> "IF-3",
    "IF-2" <-> "IF-3"
}
```

Pour des raisons de performances évidentes, les règles les plus contractées (utilisation de ALL et OTHER) seront plus faciles à traiter que leurs équivalences expansées (toutes les règles avec des <->). Néanmoins, certaines fois l'administrateur ne pourra pas contracter complètement ces règles et le collecteur devra faire avec !

3.5 La clause IF_AGGREGATION

- Optionnelle.
- Niveau : section routeur.
- Nombre : 1.
- Syntaxe : IF_AGGREGATION { IF_1 (aaa.aaa.aaa.aaa_1) [, IF_n (aaa.aaa.aaa.aaa_n) ...] }
  où : IF_i = "IfDesc SNMP"
- Paramètres :
  - IF_i étant "ifDescr SNMP" qui correspond à la description d'une interface physique ou virtuelle du routeur. Les paquets IP entrants et sortants de cette interface seront agrégés par l'adresse IP d'agrégation qui suit.
  - aaa.aaa.aaa.aaa_i : adresse IP d'agrégation des paquets IP entrants et sortants de l'interface du routeur désignée précédemment. Ex : aaa.aaa.aaa.aaa_i = 1.2.3.4.

La règle IF_AGGREGATION a essentiellement 2 objectifs :

1. elle permet d'exprimer le fait que des paquets IP sont en provenance ou à destination d'une adresse IP virtuelle (l'adresse IP d'agrégation) correspondant à une interface donnée du routeur. En

consultant les fichiers accdump (se reporter à Avant de commencer...) et les couples @IP source - @IP destination, l'adresse d'agrégation pourra apparaître en source ou destination et exprimer le fait que le trafic en question était en source ou en destination d'une interface donnée du routeur. Notons que c'est le seul moyen au niveau des fichiers accdump de savoir qu'un trafic est source ou destination d'une interface spécifique du routeur car la notion d'*ifIN* - *ifOUT* n'y est plus stockée.

2. Elle permet également de "soulager" le processus netMETacc, essentiellement au niveau de la gestion de la table @IP source - @IP destination qui peut devenir importante dans le cas de collecte non-agrégée.

Pour comprendre dans un premier temps comment fonctionne la table @IP source - @IP destination, nous allons étudier un exemple concret de réception de flows dans un contexte non-agrégé après quoi nous reprendrons le même exemple en appliquant une règle d'agrégation. La Figure 13 illustre la topologie du réseau utilisé avec :

- un routeur à 4 interfaces, toutes activées en NetFlow,
- 2 machines serveurs sur l'interface IF-0
- et 2 machines hôtes sur les interfaces IF-1 et IF-2.

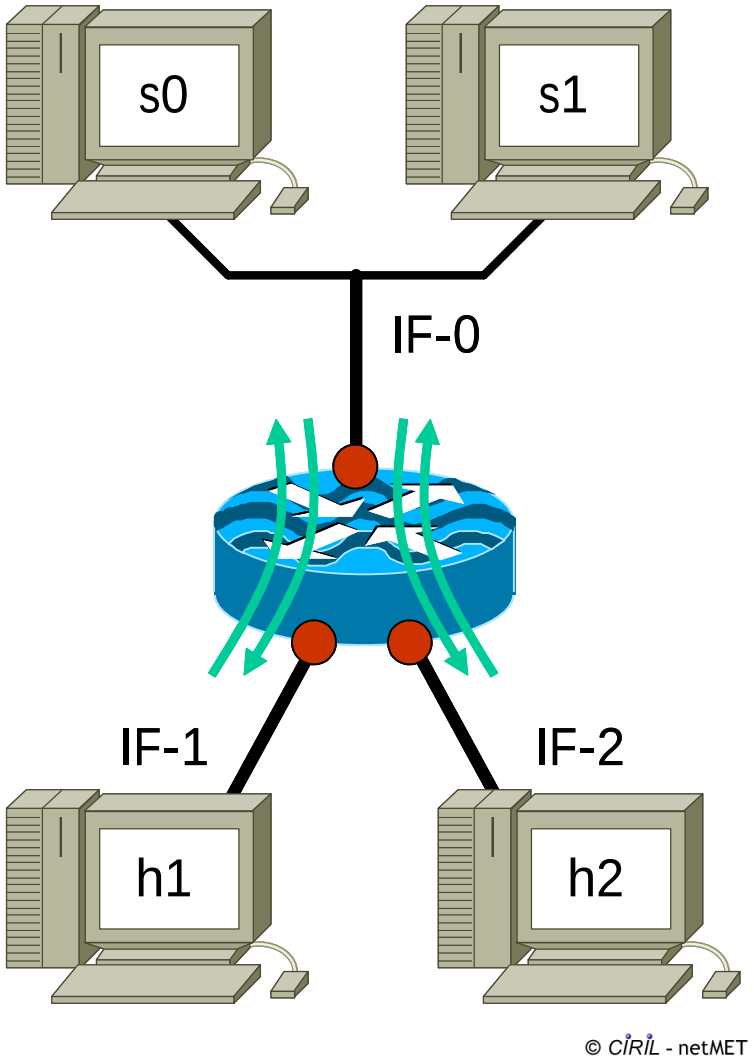

Figure 13 - Topologie pour l'exemple d'agrégation

Le tableau de gauche de la Figure 14 fait apparaître les différents flows détectés par le routeur et envoyés au processus netMETcll. Dans notre exemple, netMETcll est programmé pour ne conserver les flows qu'entre IF-0 <-> IF-1 et IF-0 <-> IF-2 (nous ne faisons pas apparaître d'autres sens de communication car ce n'est pas le sujet ici...) et ne pas appliquer d'agrégation sur les flows. Le tableau de droite représente quant à lui la table @IP source - @IP destination complète du processus netMETacc (qui reçoit les flows traités par netMETcll) à chaque réception de nouveau flow.

Flows détectés par le routeur

T	Src	Dst	Octets
0	H1	S0	10
1	S0	H1	20
2	H2	S1	30
3	H2	S1	30
4	H1	S1	10
5	S0	H1	40
6	S1	H1	40
7	H1	S0	10

Table d'accounting du processus *netMETacc*
Flows NON AGREGES

T	Src-Dst	Cmpt octets
0	H1-S0	10
1	H1-S0	10
	S0-H1	20
2	H1-S0	10
	S0-H1	20
	H2-S1	30
3	H1-S0	10
	S0-H1	20
	H2-S1	30+30=60
4	H1-S0	10
	S0-H1	20
	H2-S1	60
	H1-S1	10
5	H1-S0	10
	S0-H1	20+40=60
	H2-S1	60
	H1-S1	10
6	H1-S0	10
	S0-H1	60
	H2-S1	60
	H1-S1	10
	S1-H1	40
7	H1-S0	10+10=20
	S0-H1	60
	H2-S1	60
	H1-S1	10
	S1-H1	40

Figure 14 - Flows non-agrégés et table netMETacc

Détaillons les premières étapes du fonctionnement :

- T=0, le flow H1-> S0 arrive à netMETcll qui le passe directement à netMETacc, le couple n'existe pas encore dans la table, il est inséré (avec également le nombre d'octets de la communication entre H1 et S0, en réalité c'est un compteur d'octets par service/protocole qui est géré mais pour simplifier l'exemple nous ne prenons que des octets seuls).
- T=1, le flow S0 -> H1 arrive à netMETcll qui le passe directement à netMETacc, le couple n'existe pas encore dans la table, il est inséré.
- T=2, le flow H2 -> S1 arrive à netMETcll qui le passe directement à netMETacc, le couple n'existe pas encore dans la table, il est inséré.
- T=3, le flow H2 -> S1 arrive à netMETcll qui le passe directement à netMETacc, le couple existe déjà dans la table, on incrémente juste le compteur d'autant d'octets que pour ce nouveau flow détecté.
- ... et ainsi de suite.

On voit qu'à l'étape T=7, 5 couples source-destination ont été créés et par extension sur notre exemple le nombre maximum de couples serait nb_machines * nb_serveurs * 2 = 8. Il est clair que si l'interface IF-0 correspond à une connexion vers l'Internet le nombre de couple peut facilement augmenter...

Reprenons maintenant le même exemple, mais cette fois les flows étant agrégés par le processus netMETcll avant d'être passés à netMETacc. La règle utilisée sera : IF_AGGREGATION { "IF-0" (A) }. (Pour l'exemple, nous n'utilisons pas d'adresse IP pour l'adresse d'agrégation). La Figure 15 reprend les flows détectés et les états de la table netMETacc aux différentes étapes.

Flows détectés par le routeur

T	Src	Dst	Octets
0	H1	S0	10
1	S0	H1	20
2	H2	S1	30
3	H2	S1	30
4	H1	S1	10
5	S0	H1	40
6	S1	H1	40
7	H1	S0	10

Table d'accounting du processus *netMETacc*
Flows AGREGES avec @A

T	Src-Dst	Cmpt octets
0	H1-A	10
1	H1-A A-H1	10 20
2	H1-A A-H1 H2-A	10 20 30
3	H1-A A-H1 H2-A	10 20 30+30=60
4	H1-A A-H1 H2-A	10+10=20 20 60
5	H1-A A-H1 H2-A	20 20+40=60 60
6	H1-A A-H1 H2-A	20 60 +40=100 60
7	H1-A A-H1 H2-A	20+10=30 100 60

Figure 15 - Flows agrégés et table netMETacc

Détaillons les premières étapes du fonctionnement :

- T=0, le flow H1-> S0 arrive à netMETcll, l'adresse S0 est à destination de l'interface IF-0, netMETcll substitue donc l'adresse S0 par l'adresse d'agrégation A et passe le flow à netMETacc, le couple H1-A n'existe pas dans la table, il est inséré.
- T=1, le flow S0-> H1 arrive à netMETcll, l'adresse S0 est en provenance de l'interface IF-0, netMETcll substitue donc l'adresse S0 par l'adresse d'agrégation A et passe le flow à netMETacc, le couple A-H1 n'existe pas dans la table, il est inséré.
- T=2, le flow H2-> S1 arrive à netMETcll, l'adresse S1 est à destination de l'interface IF-0, netMETcll substitue donc l'adresse S1 par l'adresse d'agrégation A et passe le flow à netMETacc, le couple H2-A n'existe pas dans la table, il est inséré.
- T=3, le flow H2-> S1 arrive à netMETcll, l'adresse S1 est à destination de l'interface IF-0, netMETcll substitue donc l'adresse S1 par l'adresse d'agrégation A et passe le flow à netMETacc, le couple H2-A existe déjà dans la table, on incrémente juste le compteur d'autant d'octets que pour ce nouveau flow détecté.
- T=4, le flow H1-> S1 arrive à netMETcll, l'adresse S1 est à destination de l'interface IF-0, netMETcll substitue donc l'adresse S1 par l'adresse d'agrégation A et passe le flow à netMETacc, le couple H1-A existe déjà dans la table, on incrémente juste le compteur d'autant d'octets que pour ce nouveau flow détecté.
- ... et ainsi de suite.

On voit qu'à l'étape T=7, seuls 3 couples source-destination ont été créés, alors que dans l'exemple non-agrégé 5 couples étaient présents au final. Par extension sur cet exemple agrégé le nombre maximum de couples serait : nb_machines * 2 = 4.

L'intérêt de l'agrégation est évident car la taille de la table dans le cas non-agrégé est fonction des nombres de {machines et serveurs} alors que dans le cas agrégé ce n'est fonction que du nombre de {machines}. De plus dans le cas non-agrégé, après coup, on ne sait plus que les serveurs S0 et S1 se trouvaient derrière l'interface IF-0 et donc que les trafics de H1 et H2 étaient en source/destination de cette interface. Alors que l'utilisation d'une adresse spécifique d'agrégation (A) dans le 2ème exemple permet de savoir que les machines H1 et H2 ont communiqué avec l'adresse (A) qui est l'adresse d'agrégation de l'interface IF-0.

Le mécanisme d'agrégation est bien utile pour simplifier la table @IP source - @IP destination et pour, après coup, reconnaître un trafic en source/destination d'une interface particulière. Malheureusement, qui dit "agrégation" dit "contraction de l'information" et ici on voit clairement que les adresses IP des serveurs S0 et S1 "disparaissent". Donc suivant les besoins et la granularité de l'information finale voulus, il faudra faire un mixe avec agrégation et non-agrégation. C'est d'ailleurs ce qui est fait dans les 4 services netMET de la distribution (metro, stats, secure24h et secure10m) où :

- metro : est agrégé,
- stats : est agrégé,
- secure24h : est non-agrégé,
- secure10m : est non-agrégé.

4 Tips and Tricks : Choix de la "bonne" interface

Nous avons vu dans les chapitres La clause SNMP_READ_COMMUNITY, La clause IF_PROCESSED et La clause IF_AGGREGATION, que les interfaces d'un routeur sont désignées par leur description SNMP MIB-II. Le script getIF.sh permet de récupérer ces descriptions mais un problème peut se poser pour le choix de la "bonne" description en fonction de l'interface choisie...

En effet, prenons l'exemple d'un routeur avec une carte ATM qui fait de l'IP au-dessus d'ATM selon le mode RFC1483. Ce routeur, illustré par la Figure 16, a une interface ATM nommée ATM2/0 et une FastEthernet nommée FastEthernet0/0.

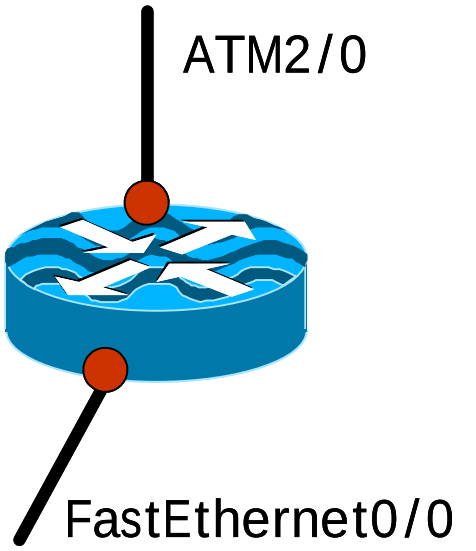

Figure 16 - Choix de la "bonne" interface : le routeur

La Figure 17 montre un exemple d'exécution du script getIF.sh sur ce routeur. On y voit apparaître les interfaces physiques : 1 et 2, l'interface loopback : 7 et les sous-interfaces logiques : 3, 4, 5, 6, 8 et 9.

```
netmet@metro [007] ~/netMet/scripts> getIF.sh mon_routeur.domain.fr
RFC1213-MIB
interfaces.ifTable.ifEntry.ifDescr.1 = ATM2/0
interfaces.ifTable.ifEntry.ifDescr.2 = FastEthernet0/0
interfaces.ifTable.ifEntry.ifDescr.3 = ATM2/0-atm layer
interfaces.ifTable.ifEntry.ifDescr.4 = ATM2/0.0-atm subif
interfaces.ifTable.ifEntry.ifDescr.5 = ATM2/0-aal5 layer
interfaces.ifTable.ifEntry.ifDescr.6 = ATM2/0.0-aal5 layer
interfaces.ifTable.ifEntry.ifDescr.7 = Null0
interfaces.ifTable.ifEntry.ifDescr.8 = ATM2/0.999-atm subif
interfaces.ifTable.ifEntry.ifDescr.9 = ATM2/0.999-aal5 layer
```


Figure 17 - Choix de la "bonne" interface : le script *getIF.sh*

Maintenant imaginons que l'on souhaite écrire une règle IF_AGGREGATION sur l'interface ATM2/0 de manière à agréger tout le trafic qui sort/entre de cette interface. Quelle description choisir ???

Une première piste peut déjà être donnée, car sur la configuration du routeur nous savons que la sous-interface qui réalise l'encapsulation RFC1483 est la sous-interface ATM2/0.999, mais le problème reste toujours entier : pourquoi ne pas choisir l'interface 1, plutôt que les autres ??? Voici comment procéder :

- le groupe d'expérimentation netMET a maintenant une certaine connaissance de cette problématique et pourrait toujours donner une indication sur l'interface à choisir,
- sinon, on peut utiliser le collecteur en mode verbose pour lui faire afficher les flows qu'il reçoit et lire les "bons" numéros d'interfaces qui sont envoyés par le routeur et ainsi faire correspondre les descriptions d'interfaces correspondantes...

Pour lancer le collecteur en mode verbose, il suffit d'utiliser les options --start (pour démarrer) et --printFLOWS (pour imprimer tous les flows reçus). La capture de la Figure 18 illustre ce démarrage en mode verbose.

```
netmet@metro [007] ~/netMet/metro> netMETcll --start --printFLOWS ; sleep 1 ; netMETcll -k
[I] - Socket opened - 192.168.200.200/8080
[I] - SEMAPHORE for synchronisation created
[I] - SHARED MEMORY created (786360 bytes)
[I] - Process attached to shared memory
[I] - Collector netMETcll started - 192.168.200.200/8080
[I] - Process attached to shared memory
192.168.200.254 {9 > 2} : xxx.xxx.xxx.xxx [    80/  6] > xxx.xxx.xxx.xxx [ 1225/  6] (40)
192.168.200.254 {9 > 2} : xxx.xxx.xxx.xxx [    80/  6] > xxx.xxx.xxx.xxx [ 2136/  6] (1500)
192.168.200.254 {9 > 2} : xxx.xxx.xxx.xxx [    80/  6] > xxx.xxx.xxx.xxx [ 3843/  6] (44)
192.168.200.254 {2 > 9} : xxx.xxx.xxx.xxx [ 3873/  6] > xxx.xxx.xxx.xxx [    80/  6] (768)
192.168.200.254 {9 > 2} : xxx.xxx.xxx.xxx [    80/  6] > xxx.xxx.xxx.xxx [ 3873/  6] (84)
192.168.200.254 {2 > 9} : xxx.xxx.xxx.xxx [ 6672/ 17] > xxx.xxx.xxx.xxx [   67/ 17] (328)
...
```

© CIRIL - netMET

Figure 18 - Démarrage du collecteur en mode verbose

Il faut bien entendu, pour démarrer netMETcll, un fichier de configuration netmet.conf dans lequel ne peuvent pas encore apparaître des descriptions d'interfaces (puisque c'est ce que l'on recherche !, on tourne en rond là !) donc on utilise la mot clé réservé ALL dans la clause IF_PROCESSED. Dans cet exemple, la machine de métrologie a pour adresse 192.168.200.200 et le routeur qui envoie les NetFlow, les envoie avec comme adresse source : 192.168.200.254.

```
NETFLOW_LISTEN_ADDR_PORT { 192.168.200.200/8080 }

192.168.200.254
{
        SNMP_READ_COMMUNITY { "public" }

        IF_PROCESSED
        { ALL }
}
```

© CIRIL - netMET

Figure 19 - Fichier de configuration pour démarrage du collecteur en mode verbose

Il apparait clairement sur la capture Figure 16 que les numéros d'interfaces utilisées sont 2 et 9, et en faisant la correspondance avec la Figure 15 on trouve que les interfaces désignées par le NetFlow sont "FastEthernet0/0" et "ATM2/0.999-aal5 layer".

Nous avons ENFIN trouvé la description à utiliser pour faire l'agrégation voulue sur l'interface ATM2/0. La Figure 20montre un exemple concret de configuration pour agréger tout le trafic RFC1483 de la sous-interface ATM2/0.999 par l'adresse "virtuelle" 1.2.3.4.

```
NETFLOW_LISTEN_ADDR_PORT { 192.168.200.200/8080 }

192.168.200.254
{
        SNMP_READ_COMMUNITY { "public" }

        IF_PROCESSED
        { "ATM2/0.999-aal5 layer" <-> OTHER }

        IF_AGGREGATION
        { "ATM2/0.999-aal5 layer" (1.2.3.4) }
}
```


Figure 20 - Choix de la "bonne" interface & Fichier de configuration final

Documentation netMET		
par :	Alexandre Simon Sébastien Morosi (maj)	Alexandre.Simon@ciril.fr Sebastien.Morosi@ciril.fr
crée le :	Mars 2001	
mise à jour le :	Mars 2003	